PATENT ABSTRACTS OF JAPAN

(11)Publication number : 10-124323 = EP0824236 US6643657
(43)Date of publication of application : 15.05.1998

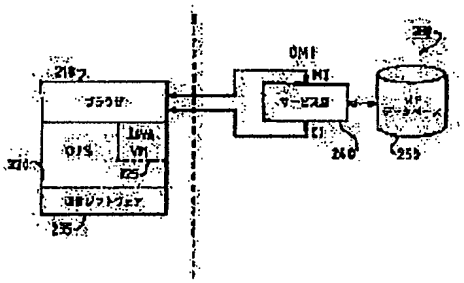
(51)Int.Cl. G06F 9/445
G06F 13/00

(21)Application number : 09-185402 (71)Applicant : INTERNATL BUSINESS MACH CORP <IBM>
(22)Date of filing : 10.07.1997 (72)Inventor : RICHARD BAILD
ALAN OGILVIE

(30)Priority
Priority number : 96 9616649 Priority date : 08.08.1996 Priority country : GB

(54) COMPUTER SYSTEM

(57)Abstract:
PROBLEM TO BE SOLVED: To provide a mechanism which makes it possible to use information regarding a code that can be accessed through a network.
SOLUTION: A computer work station is connected to a network such as the Internet. A browser (software) 210 runs on the computer work station can acquire data by accessing its page through the network and the page of the data is displayed on the computer work station side. A reference to a code can be added to the page of the data. The browser 201 can also obtain the code from the network and it can be executed on the computer work station. When the code is referred to, access can be gained through the network and the reference to another file containing information regarding the code such as the size, etc., is related. The browser 210 accesses the information in this file before acquiring or executing the code.



PF030016 US
2/5/10
Corrs To US6643657

(19)日本国特許庁（ＪＰ）　(12)公開特許公報（Ａ）　(11)特許出願公開番号

特開平10－124323

(43)公開日　平成10年(1998)５月15日

(51)Int.Cl.[6]	識別記号	ＦＩ	
Ｇ０６Ｆ　9/445		Ｇ０６Ｆ　9/06	４２０Ｊ
13/00	３５１	13/00	３５１Ｈ

審査請求　未請求　請求項の数58　ＯＬ　（全 15 頁）

(21)出願番号　特願平9－185402

(22)出願日　平成９年(1997)７月10日

(31)優先権主張番号　９６１６６４９．１

(32)優先日　1996年８月８日

(33)優先権主張国　イギリス（ＧＢ）

(71)出願人　390009531
インターナショナル・ビジネス・マシーンズ・コーポレイション
ＩＮＴＥＲＮＡＴＩＯＮＡＬ　ＢＵＳＩＮＥＳＳ　ＭＡＳＣＨＩＮＥＳ　ＣＯＲＰＯＲＡＴＩＯＮ
アメリカ合衆国10504、ニューヨーク州　アーモンク　（番地なし）

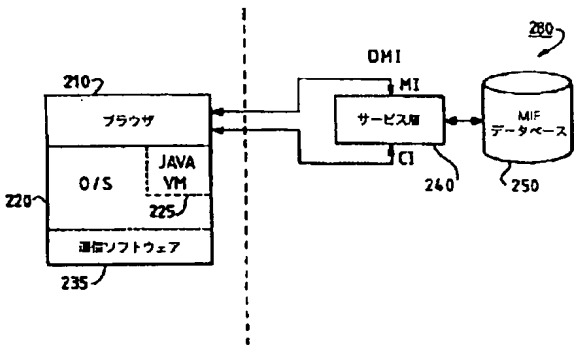
(72)発明者　リチャード・ベイルド
カナダ、エル３アール、８ブイ３、オンタリオ州マーカム、ウォーターブリッジ・レーン　94

(74)代理人　弁理士　坂口　博　（外１名）

最終頁に続く

(54)【発明の名称】　コンピュータ・システム

(57)【要約】
【課題】従って本発明は、ネットワークを通してアクセスできるコードに関する情報を使用できるようにする機構を提供する。
【解決手段】　コンピュータ・ワークステーション１がインターネット３４等のネットワークに接続される。コンピュータ・ワークステーションで実行されるブラウザ（ソフトウェア）２１０はネットワークを通してデータのページにアクセスしてこれを取得でき、データのページはコンピュータ・ワークステーション側で表示される。データのページにはコードへの参照を追加できる。コードはブラウザもネットワークから取得してコンピュータ・ワークステーションで実行することができる。コードへの参照には、ネットワークを通してアクセスでき、コードのサイズ等、コードに関する情報を含む他のファイルへの参照も関連付けられる。ブラウザはコードを取得または実行する前にこのファイルの情報にアクセスする。



【特許請求の範囲】
【請求項１】ネットワークに接続されたコンピュータ側で実行されるよう、ネットワークを通して取得できるコードへの参照を含むデータの少なくとも１ページにアクセスできる、前記コンピュータを操作する方法であって、
前記参照に関連付けられた、前記コードに関する情報の識別子を検出するステップと、
前記コードに関する情報を取得するステップと、
を含む、方法。
【請求項２】前記コードを実行するかどうかを決定するために、前記情報を利用するステップを含む、請求項１記載の方法。
【請求項３】前記コンピュータのユーザに前記情報の少なくとも一部を提示することで、前記コードを実行するかどうかを前記ユーザが決定できるようにするステップを含む、請求項２記載の方法。
【請求項４】前記情報の少なくとも一部を所定基準と比較することで、前記コードを実行するかどうかを決定するステップを含む、請求項２または３記載の方法。
【請求項５】前記データの少なくとも１ページはＨＴＭＬ形式である、請求項１乃至４のいずれかに記載の方法。
【請求項６】前記参照はＡＰＰＬＥＴタグを含む、請求項５記載の方法。
【請求項７】前記識別子は前記ＡＰＰＬＥＴタグ内の属性を含む、請求項６記載の方法。
【請求項８】前記識別子は、前記情報が置かれるサーバとディレクトリを指定する第１属性と、前記情報を含むファイルの名前を指定する第２属性とを含む、請求項７記載の方法。
【請求項９】前記コンピュータは、Ｗｅｂブラウザを実行して前記データの少なくとも１ページにアクセスし、前記参照は汎用リソース・ロケータ（ＵＲＬ）の形である、請求項１乃至８のいずれかに記載の方法。
【請求項１０】前記識別子は汎用リソース・ロケータ（ＵＲＬ）の形である、請求項１乃至９のいずれかに記載の方法。
【請求項１１】前記情報は、管理情報形式（ＭＩＦ）に準拠した形で保存される、請求項１乃至１０のいずれかに記載の方法。
【請求項１２】前記情報はファイルに保存され、前記コードに関する情報を取得するステップは、前記ファイルを前記コンピュータにコピーするステップを含む、請求項１乃至１１のいずれかに記載の方法。
【請求項１３】前記情報は、管理情報形式（ＭＩＦ）に準拠した形で保存され、前記コンピュータ側のＭＩＦデータベースにロードされる、請求項１２記載の方法。
【請求項１４】ロードされたＭＩＦ情報にアクセスするため前記ＭＩＦデータベースに問い合わせをするステップを含む、請求項１３記載の方法。
【請求項１５】前記情報はリモート・データベースに保存され、前記情報を取得するステップは、ネットワークを通して前記データベースにリモートで問い合わせをするステップを含む、請求項１乃至１１記載の方法。
【請求項１６】前記情報は前記コードのサイズを含む、請求項１乃至１５のいずれかに記載の方法。
【請求項１７】ネットワークを通して前記コードを取得するため前記参照を使用するステップを含む、請求項１乃至１６のいずれかに記載の方法。
【請求項１８】取得されたコードを前記コンピュータ側で実行するステップを含む、請求項１７記載の方法。
【請求項１９】ネットワークに接続でき、コンピュータ側で実行するため、前記ネットワークを通して取得できるコードへの参照を含むデータの少なくとも１ページにアクセスするコンピュータ・システムであって、
前記参照に関連付けられた、前記コードに関する情報の識別子を検出する手段と、
前記コードに関する情報を取得する手段と、
を含む、コンピュータ・システム。
【請求項２０】前記コードを実行するかどうかを決定するために、前記情報を利用する手段を含む、請求項１９記載のコンピュータ・システム。
【請求項２１】前記コンピュータのユーザに前記情報の少なくとも一部を提示することで、前記コードを実行するかどうかを前記ユーザが決定できるようにする手段を含む、請求項２０記載のコンピュータ・システム。
【請求項２２】前記情報の少なくとも一部を所定基準と比較することで、前記コードを実行するかどうかを決定する手段を含む、請求項２０または２１記載のコンピュータ・システム。
【請求項２３】前記データの少なくとも１ページはＨＴＭＬ形式である、請求項１９乃至２２のいずれかに記載のコンピュータ・システム。
【請求項２４】前記参照はＡＰＰＬＥＴタグを含む、請求項２３記載のコンピュータ・システム。
【請求項２５】前記識別子は、前記ＡＰＰＬＥＴタグ内の属性を含む、請求項２４記載のコンピュータ・システム。
【請求項２６】前記識別子は、前記情報が置かれるサーバとディレクトリを指定する第１属性と、前記情報を含むファイルの名前を指定する第２属性とを含む、請求項２５記載のコンピュータ・システム。
【請求項２７】前記コンピュータは、Ｗｅｂブラウザを実行して前記データの少なくとも１ページにアクセスし、前記参照は汎用リソース・ロケータ（ＵＲＬ）の形である、請求項１９乃至２６のいずれかに記載のコンピュータ・システム。
【請求項２８】前記識別子は汎用リソース・ロケータ（ＵＲＬ）の形である、請求項１９乃至２７のいずれか

に記載のコンピュータ・システム。
【請求項２９】前記情報は、管理情報形式（ＭＩＦ）に準拠した形で保存される、請求項１９乃至２８のいずれかに記載のコンピュータ・システム。
【請求項３０】前記情報はファイルに保存され、前記コードに関する情報を取得する手段は、前記ファイルを前記コンピュータにコピーする手段を含む、請求項１９乃至２９のいずれかに記載のコンピュータ・システム。
【請求項３１】前記情報は、管理情報形式（ＭＩＦ）に準拠した形で保存され、前記コンピュータ側のＭＩＦデータベースにロードされる、請求項３０記載のコンピュータ・システム。
【請求項３２】ロードされたＭＩＦ情報にアクセスするため前記ＭＩＦデータベースに問い合わせをする手段を含む、請求項３１記載のコンピュータ・システム。
【請求項３３】前記情報はリモート・データベースに保存され、前記情報を取得する手段は、ネットワークを通して前記データベースにリモートで問い合わせをする手段を含む、請求項１９乃至２９のいずれかに記載のコンピュータ・システム。
【請求項３４】前記情報は前記コードのサイズを含む、請求項１９乃至３３のいずれかに記載のコンピュータ・システム。
【請求項３５】ネットワークを通して前記コードを取得するため前記参照を使用する手段を含む、請求項１９乃至３４のいずれかに記載のコンピュータ・システム。
【請求項３６】取得されたコードを前記コンピュータ側で実行する手段を含む、請求項３５記載のコンピュータ・システム。
【請求項３７】ネットワークに接続されたコンピュータによって実行され、前記コンピュータ側で実行するためにネットワークを通して取得できるコードへの参照を含むデータの少なくとも１ページにアクセスする、コンピュータ・プログラム・プロダクトであって、
前記参照に関連付けられた、前記コードに関する情報の識別子を検出する手段と、
前記コードに関する情報を取得する手段と、
を含む、コンピュータ・プログラム・プロダクト。
【請求項３８】前記コードを実行するかどうかを決定するために、前記情報を利用する手段を含む、請求項３７記載のコンピュータ・プログラム・プロダクト。
【請求項３９】前記コンピュータのユーザに前記情報の少なくとも一部を提示することで、前記コードを実行するかどうかを前記ユーザが決定できるようにする手段を含む、請求項３８記載のコンピュータ・プログラム・プロダクト。
【請求項４０】前記情報の少なくとも一部を所定基準と比較することで、前記コードを実行するかどうかを決定する手段を含む、請求項３８または３９記載のコンピュータ・プログラム・プロダクト。
【請求項４１】前記情報はファイルに保存され、前記コードに関する情報を取得する手段は、前記ファイルを前記コンピュータにコピーする手段を含む、請求項３７乃至４０のいずれかに記載のコンピュータ・プログラム・プロダクト。
【請求項４２】前記情報は、管理情報形式（ＭＩＦ）に準拠した形で保存され、前記コンピュータ側のＭＩＦデータベースにロードされる、請求項４１記載のコンピュータ・プログラム・プロダクト。
【請求項４３】ロードされたＭＩＦ情報にアクセスするため前記ＭＩＦデータベースに問い合わせをする手段を含む、請求項４２記載のコンピュータ・プログラム・プロダクト。
【請求項４４】前記情報はリモート・データベースに保存され、前記情報を取得する手段は、ネットワークを通して前記データベースにリモートで問い合わせをする手段を含む請求項３７乃至４０のいずれかに記載のコンピュータ・プログラム・プロダクト。
【請求項４５】ネットワークを通して前記コードを取得するため前記参照を使用する手段を含む、請求項３７乃至４４のいずれかに記載のコンピュータ・プログラム・プロダクト。
【請求項４６】取得されたコードを前記コンピュータ側で実行する手段を含む、請求項４５記載のコンピュータ・プログラム・プロダクト。
【請求項４７】インターネットを通して取得でき、コンピュータ側で実行されるアプレット・コードへの参照を含む、ワールド・ワイド・ウェブ上のデータの少なくとも１ページにアクセスするＷｅｂブラウザであって、
前記参照に関連付けられた、前記アプレット・コードに関する情報の識別子を検出する手段と、
前記コードに関する情報を取得する手段と、
を含む、ブラウザ。
【請求項４８】前記コードを実行するかどうかを決定するために、前記情報を利用する手段を含む、請求項４７記載のブラウザ。
【請求項４９】前記コンピュータのユーザに前記情報の少なくとも一部を提示することで、前記コードを実行するかどうかを前記ユーザが決定できるようにする手段を含む、請求項４７記載のブラウザ。
【請求項５０】前記情報の少なくとも一部を所定基準と比較することで、前記コードを実行するかどうかを決定する手段を含む、請求項４８または４９記載のブラウザ。
【請求項５１】前記参照はＡＰＰＬＥＴタグを含み、前記識別子は前記ＡＰＰＬＥＴタグ内の属性を含む、請求項４７乃至５０のいずれかに記載のブラウザ。
【請求項５２】前記識別子は、前記情報が置かれるサーバとディレクトリを指定する第１属性と、前記情報を含むファイルの名前を指定する第２属性とを含む、請求項

５１記載のブラウザ。
【請求項５３】前記情報は、管理情報形式（ＭＩＦ）に準拠した形で保存される、請求項４７乃至５２のいずれかに記載のブラウザ。
【請求項５４】前記情報はファイルに保存され、前記コードに関する情報を取得する手段は、前記ファイルを前記コンピュータにコピーする手段を含む、請求項４７乃至５３のいずれかに記載のブラウザ。
【請求項５５】前記情報は、管理情報形式（ＭＩＦ）に準拠した形で保存され、前記コンピュータ側のＭＩＦデータベースにロードされる、請求項５４記載のブラウザ。
【請求項５６】ロードされたＭＩＦ情報にアクセスするため前記ＭＩＦデータベースに問い合わせをする手段を含む、請求項５５記載のブラウザ。
【請求項５７】前記情報はリモート・データベースに保存され、前記情報を取得する手段は、ネットワークを通して前記データベースにリモートで問い合わせをする手段を含む請求項４７乃至５３のいずれかに記載のブラウザ。
【請求項５８】ネットワークを通して前記コードを取得し、取得されたコードを前記コンピュータ側で実行する手段を含む、請求項４７乃至５７のいずれかに記載のブラウザ。
【発明の詳細な説明】
【０００１】
【発明の属する技術分野】本発明は、コンピュータ・システム及びコンピュータ・システムからネットワークを通してデータやコードにアクセスできるように、ネットワークに接続されたコンピュータ・システムを操作する方法に関する。
【０００２】
【従来の技術】過去数年間、インターネット、特にインターネットの最上部に置かれる機構の１つであるＷＷＷ（ワールド・ワイド・ウェブ）が急激に拡大している。ＷＷＷは、多くの異種サーバに分散した多数の情報ページまたはファイルで構成される。各ページはＵＲＬ（汎用リソース・ロケータ）で識別される。ＵＲＬはサーバ・マシン及びそのマシン上の特定のファイルまたはページの両方を示す。１つのサーバに多くのページやＵＲＬが存在可能である。
【０００３】クライアントがＷＷＷを使用するときには、WebExplorer（ＩＢＭのオペレーティング・システムＯＳ／２の一部）、NetscapeのNavigatorプログラム等、Ｗｅｂブラウザと呼ばれるソフトウェアを実行する。クライアントはブラウザと対話し、特定のＵＲＬを選択する。ＵＲＬによりブラウザは、そのＵＲＬまたはページに対する要求をＵＲＬで識別されたサーバに送る。通常、サーバは要求に応えるため要求されたページを取得し、そのページのデータを要求側クライアントに転送する（クライアントとサーバの対話は、ＨＴＴＰ（ハイパーテキスト・トランスポート・プロトコル）に従って行われる）。このページは次にクライアントの画面に表示される。また例えば特定のテーマに関係したＷＷＷページを検索するため、クライアントによりサーバがアプリケーションを起動することもできる。
【０００４】ほとんどのＷＷＷページは、ＨＴＭＬ（ハイパーテキスト・マークアップ言語）と呼ばれる言語に従ってフォーマットされる。従って、通常のページにはテキストと共にタグと呼ばれるフォーマット・コマンドが埋め込まれる。タグを使用してフォントの大きさ、フォントのスタイル（イタリックかボールドか等）、テキストのレイアウト方法等が制御される。Ｗｅｂブラウザは指定フォーマットに従ってテキストを表示するためにＨＴＭＬスクリプトを解析する。またＨＴＭＬページには、他のＵＲＬの形で画像、ビデオ・セグメント、オーディオ・ファイル等、マルチメディア・データへの参照を加えることができる。Ｗｅｂブラウザはこのような参照に応答するため、データを取得して表示するかまたは再生する。このようなマルチメディア・データで専用のＷＷＷページを周囲のＨＴＭＬテキストを使用せずに作成することも可能である。
【０００５】ほとんどのＷＷＷページには、元のページと同じサーバにある必要のない、他のＷＷＷページへの参照もある。こうした参照は、一般にはユーザが画面の特定の位置を選択することによって起動される。通常はマウス制御ボタンをクリック（ダブルクリック）する。これらの参照またはロケーションはハイパーリンクとよばれ、通常、ブラウザにより特定の形でフラグがたてられる（例えばハイパーリンクに関連付けられたテキストの色を変えることができる）。ユーザがハイパーリンクを選択すると、参照されたページが取得され、現在表示されているページに代わる。
【０００６】この他、ＨＴＭＬやＷＷＷについては、Dr Dobbs Journal、December 1994のDouglas McArthurによる"World Wide Web and HTML"、及びIan Grahamによる"TheHTML SourceBook"（John Wiley、New York、1995）を参照されたい。
【０００７】ここまで述べたことから言えば、また現在実現されているものとして概要に言及すると、ＷＷＷにはサーバからクライアントにダウンロードされたページは基本的に受動的である。言い換えると、クライアント・マシン側で実行されるコードを持たないという欠点がある。これが意味することは、１つには、サーバはクライアントとサーバの対話（インタラクション）に関連した処理をクライアントにオフロードできないということである。従って、クライアントが例えば電話番号で書式を埋める場合、電話番号の桁数等の正規のチェックは、サーバ側で行う必要がある。その場合、まずサーバ側で処理の負荷が重くなり、次に修正すべきエラーが生じた

場合には、サーバとクライアントの間で時間のかかる余計な通信が生じる。更にサーバがクライアント側で実行されるコードをダウンロードできないことは、WWWを利用するために開発されたアプリケーションには大きな制限になる。

【０００８】最近開発されたもの、特にSun MicrosystemsのＪａｖａ技術をもとにしたものは、上に述べた問題を克服することを課題にしている。Ｊａｖａ技術は主として、ｉ）いくらかCやC++に似た新しいプログラミング言語と、ｉｉ）仮想マシンで構成される。基本的にＪａｖａプログラミング言語で書かれたプログラムは、バイト・コード形式にコンパイルし、次にＪａｖａ仮想マシン上で実行時に解釈することができる。Ｊａｖａ仮想マシンはバイト・コードをその物理マシンによって実行できる命令に変換する。

【０００９】Ｊａｖａで書かれたプログラムは、クライアント側のＪａｖａ仮想マシンで実行されるバイト・コードの形でWWWを通してダウンロードできる。このようなプログラムは"アプレット（applet）"と呼ばれる。Ｊａｖａ技術によりWWWを通してコードをダウンロードすることには大きな利点が２つある。まず、アプレットはプラットフォームに依存しない。ただしこれは各クライアントにＪａｖａ仮想マシンのコピーがあると想定する場合に限られる（クライアントのシステム側の仮想マシンは、通常はオペレーティング・システムに組み込まれているか、Ｗｅｂブラウザ自体に組み込まれている）。言い換えると、サーバはそのオペレーティング・システムやマシンに従って、クライアントにダウンロードするため異なるコードのバージョンを持つ必要がない。従って、１つのバージョンの関連コードだけを書込み、維持すればよく、これはソフトウェア開発者には都合がよい。第２点として、アプレットは物理マシンではなく仮想マシンで実行されるのでセキュリティが大幅に向上する。従って、ネットワークからコードをダウンロードするときには、クライアント側に保存されたデータやプログラムに損害を与える有害コードが（偶発的にか他の形で）含まれるというリスクが常にある。しかし仮想マシンはアプレットの動作を監視できるので、そうした有害な操作を検出し防止することができる。

【００１０】ソフトウェアを仮想マシンで実行するために、バイト・コードの形でサーバからクライアントにダウンロードするという考え方は、Ｊａｖａ技術とはまた別の形でも知られていることに注意されたい（例えば米国特許番号第５３４７６３２号を参照）。

【００１１】Ｊａｖａアプレットを起動するため、ＨＴＭＬテキストのＷｅｂページに＜ＡＰＰＬＥＴ＞タグが含まれる。このタグは、アプレットを含むＵＲＬを識別する。ブラウザはこのタグに応答するためアプレットを取得して実行する。また＜ＰＡＲＡＭ＞タグも定義される。これは対応する＜ＡＰＰＬＥＴ＞と＜／ＡＰＰＬＥＴ＞のタグのペアの中に保持され、実行時にアプレットに渡されるパラメータを指定するのに使用できる。（ＡＰＰＬＥＴタグとＰＡＲＡＭタグは、ＨＴＭＬ規格に正式には取り入れられていないが、それでも多くのＷｅｂブラウザにより認識される）。Ｊａｖａ技術やアプレットの詳細については、Laura LemayとCharles Perkinsによる"Teach Yourself Java in 21 Days"（Sams. net Publishing、Indianapolis、USA、1996）を参照されたい。

【００１２】前記のアプローチの問題は、ＡＰＰＬＥＴタグ内に保持できる情報の量が制限されることである。これはまた別の問題につながることがあるが、それはＨＴＭＬページ内で参照された特定のアプレットをダウンロードするかどうかをブラウザが決定しなければならない場合である。例えばブラウザは特別に大きい（従ってネットワークで転送すると遅くなる）か、或いはクライアント側に現在インストールされているものより新しいバージョンのＪａｖａ仮想マシンを必要とする、アプレットはダウンロードしようとしないことがある。

【００１３】従って本発明は、ネットワークに接続されたコンピュータを操作し、前記コンピュータ側で実行するために、ネットワークを通して取得できるコードへの参照を含むデータの少なくとも１ページへのアクセスを可能にし、前記参照に関連付けられた、前記コードに関する情報の識別子を検出するステップと、コードに関する情報を取得するステップとを含むことを特徴とする方法を提供する。

【００１４】従ってコンピュータは、識別子によりコードに関する情報を検出して取得することができる。この情報は次に、前記コードを実行するかどうかを決定するために使用できる。例えば情報は、好適にはコードの大きさを示し、コードが大きすぎて相応の時間内にダウンロードできないことを決定することができる。もう１つのの可能性として、この情報はクライアント・ワークステーション側にはインストールできない、ビデオ、キャプチャ機構等のハードウェア支援機能がコードを実行するために必要になることを指定する。このような決定は明示的に、ユーザに提示するか、所定基準をもとに自動的に実行することもできる（或いは自動的な意志決定とユーザ主導による意志決定の両方を採用することもできる）。

【００１５】この他、コードに関する情報の使用方法として考えられるのは、コードの使用にいくらかの支払いが伴う等、監視を目的にしたものや、著作権表記やライセンス契約条件の表記にアクセスできるようにする等、単に情報提供を目的としたもの等がある。更にコード自体の起動を補助することも考えられる。例えばコードが暗号化されている場合、情報によりコードを起動する前にコードの解読に必要なアルゴリズムを指定することができる。

【００１６】好適な実施例では、コードに関する情報が一度取得され、必要に応じて保存されたり処理されたりした場合は、コード自体がネットワークを通して取得される。これに代えて、コードはコードに関する情報の取得より先に取得することもできる。

【００１７】好適な実施例では、データの少なくとも１ページはＨＴＭＬ形式であり、参照はＡＰＰＬＥＴタグを含み、識別子は前記ＡＰＰＬＥＴタグ内の属性を含む。前記参照と前記識別子は、好適にはＵＲＬ（汎用リソース・ロケータ）の形をとり、参照と識別子はそれぞれ２つの部分で指定される。最初の部分はサーバとディレクトリを、次の部分は実際のファイル名を指定する。この好適な実施例の場合、アプレットはクライアント・ワークステーション側のＪａｖａ仮想マシンで実行するため、Ｊａｖａバイト・コード形式でダウンロードされる。これは、コンピュータでＷｅｂブラウザが実行され、前記少なくとも１ページのデータ、前記コード参照及び前記情報は全てインターネットからアクセスできるという典型的な状況である。

【００１８】特定のＡＰＰＬＥＴタグは特にＪａｖａ言語に関連付けられるので、今後の何らかの規格で、一般化を進めたタグ（すなわち言語依存性が少ないもの）に代わる可能性があることがわかる。もちろんこれは本発明の原理には影響を与えない。また状況によっては、前記データの少なくとも１ページはコードの参照と、これに関連する情報の識別子の他には何も含まない、或いはほとんど何も含まない可能性があることも理解されよう（言い換えると、ページの基本的目的または唯一の目的は、コードをアクセスすることとも考えられる）。

【００１９】好適な実施例の場合、情報はＭＩＦ（管理情報形式）に準拠する。これは周知の規格であり、そのためコードにアクセスしようとするクライアントにより認識される可能性が高い。しかし情報はブラウザが認識できアクセスできる任意の形式で保存することも可能である。ＭＩＦ情報はネットワークを通してコピーされ、好適には、コンピュータ側のＭＩＦデータベースにロードされる。そこで情報はブラウザにより問い合わせをすることができる（或いは単に監視を目的に保存する）。これとは別に、ＭＩＦ情報はリモートのＭＩＦデータベースに保存されていてもよい。その場合、コードに関するその情報はネットワークを通して前記データベースにリモートで問い合わせをすることにより取得できる。これによりネットワークでＭＩＦファイル全体を転送する必要がなくなる。

【００２０】

【発明が解決しようとする課題】従って本発明は、ネットワークを通してアクセスできるコードに関する情報を使用できるようにする機構を提供する。本発明による方法に代わる方法としては、例えばコードに関係した情報をＨＴＭＬページ自体に用意することが考えられる。これは例えば＜ＡＰＰＬＥＴ＞タグの属性として追加できる。ただし、そのためにタグがかなり大きくなる可能性があり、それだけでなく特定のアプレットを参照した各ページが、この情報のコピーを保持しなければならない結果にもなる。これは必要な記憶域の面から高コストになるだけでなく、実際的な観点から言えばメンテナンスが困難になる。アプレットの変更は複数の場所で反映しなければならないからである（確かにコードの所有者は、そのコードを参照するサイトの全てを意識することは普通ない）。もう１つ考えられるのは、コードに関する情報をコード自体の中で識別することである。そうすればコードは実行時に、最初に関連情報を与える。この方法にはしかし、コードは少なくとも部分的には常に実行されていなければならないという欠点がある。従って、ネットワークを通してアクセスできるコードに関する情報を使用できるようにする上では、本発明によって採用される方法が最も強力で柔軟性があり、効率的な方法であることがわかる。

【００２１】

【課題を解決するための手段】本発明はまた、ネットワークに接続できるコンピュータ側で実行するために、ネットワークを通して取得できるコードへの参照を含むデータの少なくとも１ページにアクセスするコンピュータ・システムを提供する。コンピュータ・システムは前記参照に関連付けられた、前記コードに関する情報の識別子を検出する手段と、コードに関する情報を取得する手段とを含むことを特徴とする。

【００２２】本発明はまた、コンピュータ・システム側で実行されるようネットワークを通して取得できるコードへの参照を含むデータの少なくとも１ページにアクセスするため、ネットワークに接続されたコンピュータ・システムにより実行されるコンピュータ・プログラム・プロダクトを提供する。前記コンピュータ・プログラム・プロダクトは前記参照に関連付けられた、前記コードに関する情報の識別子を検出する手段と、コードに関する情報を取得する手段とを含む。

【００２３】通常、コンピュータ・プログラム・プロダクトは、ディスケットやＣＤ－ＲＯＭに置かれるか、またはネットワークでのダウンロードを目的にサーバ・マシンのテープやハード・ディスクに保存でき、使用時にはコンピュータ・システムの作業メモリ（通常はＲＡＭ）にコピーされる。

【００２４】本発明はまた、前記コンピュータ側で実行されるよう、インターネットを通して取得できるアプレット・コードへの参照を含む、ＷＷＷのデータの少なくとも１ページにアクセスするＷｅｂブラウザを提供する。ブラウザは、前記参照に関連付けられた、前記アプレット・コードに関する情報の識別子を検出する手段と、コードに関する情報を取得する手段とを含むことを特徴とする。

【００２５】
【発明の実施の形態】図１は、システム装置１０、表示画面１２、キーボード１４及びマウス１６を含むコンピュータ・システム１の概要図である。システム装置はマイクロプロセッサ（μＰ）２２、半導体メモリ（ＲＯＭ／ＲＡＭ）２４、ハード・ディスク・ドライブ２８、及びデータが転送されるバス２６を含む。ネットワーク・アダプタ・カード３０は、リンク３２と共にコンピュータ・ワークステーションがインターネット３４とデータを交換できるようにするものである。通常、コンピュータは基本的にはハード・ディスク・ドライブから、またはネットワークからＲＯＭ／ＲＡＭにロードされたソフトウェア命令をもとに、マイクロプロセッサによって制御される。図１のシステムは、ＩＢＭのパーソナル・コンピュータ、RISCSystem/6000コンピュータ等、従来のワークステーションでよい。このようなシステムについては周知の通りであり、当業者であれば、様々なシステム構成を考えることができよう。例えばＣＤ－ＲＯＭを追加し、複数のネットワーク・アダプタ・カードを使用し、ネットワークに接続するためネットワーク・アダプタ・カードの代わりにモデムを使用する、オーディオ装置を加える等が可能である。こうした変形は本発明の理解には無関係である。
【００２６】図２は、オペレーティング・システム２２０、Ｗｅｂブラウザ２１０（ブラウザはオペレーティング・システムに組み込まれている場合もある）等、システム１で実行されるソフトウェア要素を示す。先に述べたように、ブラウザはインターネットで情報等にアクセスし、取得したものを表示したり処理したりするためのものである。ブラウザで重要な機能はＨＴＭＬコマンドを認識する機能である。ＩＢＭのＡＩＸ、ＯＳ／２等のオペレーティング・システム２２０は、ワークステーションの一般的なシステム管理を担う。オペレーティング・システムには、ワークステーションがインターネット３４と対話できるようにする通信ソフトウェア２３５が含まれる。周知の通り、このような通信のベースはＴＣＰ／ＩＰプロトコルである。オペレーティング・システムはまたＪａｖａ仮想マシン（ＶＭ）２２５を含む。仮想マシン２２５はＪａｖａアプリケーションを実行するために、Ｊａｖａバイト・コードを解釈することができる。Ｊａｖａ仮想マシンはブラウザの一部として提供される場合もある。システム１では、実施例によってはデスクトップ管理インターフェイス（ＤＭＩ）・ソフトウェア２８０も実行される（これもオペレーティング・システム２２０に組み込まれることがある）。詳しくは後述する。
【００２７】システム１とそのソフトウェアはＨＴＴＰプロトコルを使用し、これまでの方法で、Ｊａｖａアプレットを含めてWWWページにアクセスし処理するために使用できる。Ｊａｖａアプレットを参照するＷｅｂページのＨＴＭＬテキストの一例をここに示す。
【数１】

```
<APPLET CODEBASE = "http://ncc.hursley.ibm.com/javainfo/classes"
    CODE = "Map.class"
    WIDTH = 560
    HEIGHT = 560
>
<param name = IMAGE value = "graphics/cup-bun.jpg">
<param name = MAP value = "java.map">
<param name = TYPE value = "NCSA">
</APPLET>
```

【００２８】例１：ブラウザはこのＨＴＭＬテキストを次のように解釈する。＜ＡＰＰＬＥＴ＞タグと＜／ＡＰＰＬＥＴ＞タグに応答し、サーバ側の"http://ncc. hursley. ibm. com/javainfo/classes"によって指定されたディレクトリのファイル"Map. class"に置かれたコードを取得する。（ＨＴＭＬでは周知の通り"＜／ｘｘｘ＞"の形のタグは、その前の対応する＜ｘｘｘ＞タグに関連付けられた情報やデータの終わりを示す。）この処理は、基本的にはＵＲＬのWWWページのアクセスと同じである。ブラウザは次にアプレット・コンテント・オブジェクトを作成する。これにＨＥＩＧＨＴとＷＩＤＴＨ（属性という）の値が関連付けられる。これらは、アプレットの実行に必要な画面の量、及び特定のパラメータ値を定義する。これらのパラメータ値は、それぞれの＜ＰＡＲＡＭ＞タグに列挙される。＜ＰＡＲＡＭ＞タグはタグの名前と値を指定し、ＪａｖａのgetParameterメソッドを使用することでアプレット自体に取り込むことができる。従って、例えばＴＹＰＥは、値"ＮＣＳＡ"が割り当てられるMap. classアプレットの文字列変数の名前である。最後にアプレットは、先にアプレット・コンテント・オブジェクトに保存されていたパラメータ・データを使用してアプレット・オブジェクトを作成してからＪａｖａのinit()命令を使用してアプレット・オブジェクトを起動することによって実行される。
【００２９】先に述べた通り、これまでの方法ではいくつか問題が生じる。例えばＪａｖａアプレットには、クライアントにインストールされたものよりも新しいバージョンのＪａｖａ仮想マシンが必要になることがある。このような状況では、クライアントはアプレットをダウンロードして実行しようとすると、アプレットの操作は

制限されている、或いは不可能であることを認識することがある。

【００３０】本発明は、この問題を克服するために、アプレットに関してブラウザに与えられる情報を拡張する。従って本発明により、例１のＨＴＭＬテキストは次のように変更される。

【数２】

```
<APPLET CODEBASE = "http://ncc.hursley.ibm.com/javainfo/classes"
    CODE = "Map.class"
    MIFBASE = "http://ncc.hursley.ibm.com/javainfo/mifs"
    MIF = "Map.mif"
    WIDTH = 560
    HEIGHT = 560
>
<param name = IMAGE value = "graphics/cup-bun.jpg">
<param name = MAP value = "java.map">
<param name = TYPE value = "NCSA">
</APPLET>
```

【００３１】例２：例2は、＜ＡＰＰＬＥＴ＞タグに２つの新しい属性が追加されていることを除いて例１と同じである点に注意されたい。２つの属性、ＭＩＦＢＡＳＥとＭＩＦはそれぞれサーバ／ディレクトリとファイルを含むＵＲＬを指定する点で、フィールドＣＯＤＥＢＡＳＥとＣＯＤＥに似ている。ただし、ＣＯＤＥＢＡＳＥとＣＯＤＥはＪａｖａアプレットの実際のコードを検出するのに対して、ＭＩＦＢＡＳＥとＭＩＦはアプレットを記述した情報を持つファイルを検索する。このファイルはアプレット・コード自体と同じサーバに、また同じディレクトリにも置け、或いは全く別のサーバに置いてもよい。

【００３２】好適な実施例の場合、アプレットを記述した情報を含むファイルは、管理情報形式（ＭＩＦ）と呼ばれる特別な形式である。これはデスクトップ管理インターフェイス（ＤＭＩ）の一部としてデスクトップ管理タスクフォース（DeskTop Management TaskForce）というコンソーシアムにより開発された標準形式である。デスクトップ管理タスクフォースとＭＩＦの概要については、Daniel Begunによる"The DMI Waiting Game"（Computer Shopper、n8、v15、p608、August 1995）及びBarbara Genglerによる"The End of the Balancing Act"（Internetwork、n8、v6、p47、August 1995）を参照されたい。ＭＩＦに関する詳細はデスクトップ管理タスクフォース（米オレゴン州ヒルズバラ）やデスクトップ管理タスクフォースのＷｅｂサイト（特に次を参照）から得られる。

【数３】

http://wwww.dmtf.org/general_info/factsheet.html、及び
http://wwww.dmtf.org/techlinks/white_papers.html

【００３３】簡単に説明すると、プレーン・テキストの形のＭＩＦファイルは複数のグループで構成される。特別な標準グループが定義されており、いくつかは必須項目であるが、ユーザが定義したグループも受け入れられる。各グループの中に名前付き属性とその値のリストがある。例えば必要なグループとして"ComponentID"があり、これは製造者、製品、バージョン、シリアル番号、インストール（日付と時刻）及び確認（当該要素が実際に存在し動作するかどうか）の各属性を持つ。図２に戻ると、ＤＭＩ２８０をサポートするシステムにはＭＩＦデータベース２５０と、サービス層２４０がある（これらはマシンのオペレーティング・システムの一部として提供することができる）。管理可能なプロダクト（アプリケーション・ソフトウェア等）は、それぞれのＭＩＦファイルをサービス層のコンポーネント・インターフェイス（ＣＩ）を介してＭＩＦデータベースに追加或いはデータベースのＭＩＦファイルを更新する。一方、管理アプリケーションはＭＩＦデータベースにサービス層の管理インターフェイス（ＭＩ）を介して管理可能なアプリケーションに関する情報を問い合わせることができる。ソフトウェアのＭＩＦファイルの例と資料は、次のＷｅｂサイトから入手することができる。

【数４】

http://www-uk.hpl.hp.com/people/arp/dmtf/ver2.htm

【００３４】ある簡単な実施例での本発明の動作を図３に示している。最初、ブラウザ・ソフトウェアが＜ＡＰＰＬＥＴ＞タグの存在を検出する（ステップ３１０）。このタグはダウンロードされ実行されるアプレットへの参照を示す。次にブラウザは、参照されたアプレットに関連付けられたＭＩＦＢＡＳＥ、ＭＩＦの各属性を探す（ステップ３２０）。これらが検出されたとするとブラウザは次に、指定されたＭＩＦファイルにアクセスして取得する（ステップ３３０、３４０）。ＭＩＦファイルは次にウィンドウに表示される（ステップ３５０）。ここでユーザはＭＩＦ情報をスクロールでき、ダイアログ・ボックスでアプレットの操作に進むかどうか聞かれる（ステップ３６０）。ＭＩＦ情報が受け入れ可能とするとユーザはそこでアプレットをダウンロードして起動することができる（ステップ３７０、３８０）。しかしア

プレットが大きすぎる、或いはＪａｖａ仮想マシンにはクライアントにインストールされたものより新しいバージョンが必要である等、ＭＩＦファイルに問題のあることがわかった場合、ユーザは操作を中断するオプションを選択することができる。これによりアプレットを不必要にダウンロードしたり実行を試したりすることがなくなる。

【００３５】ステップ３２０に戻るが、ＭＩＦ、ＭＩＦＢＡＳＥの各属性が検出されない場合は、アプレットを直接ダウンロードし起動するための処理が行われる。これは、ＭＩＦ、ＭＩＦＢＡＳＥの各属性を認識しないブラウザによって取られるパスであり、この例は従来の技術のブラウザに見られる。

【００３６】図３に示した方法は比較的に簡単であるが、ユーザがＭＩＦファイルそのものを走査しなければならないという欠点がある。より複雑な処理方法が図４の実施例に示してある（この場合、ＡＰＰＬＥＴタグにはＭＩＦ／ＭＩＦＢＡＳＥ属性がないことを前提にしている）。ＭＩＦ情報がサーバから取得されると（ステップ４０５）、サービス層のコンポーネント・インターフェイス（ＣＩ）を介してコンピュータ・システム（４１０）のＭＩＦデータベースにロードされる。次に、ＭＩＦファイルからＭＩＦ情報を読出すためにサービス層の管理インターフェイス（ＭＩ）が用いられる（ステップ４１５）。ブラウザとＤＭＩの間のこの対話は図２に概要を示している。

【００３７】ブラウザは、所定の基準に照らしてＭＩＦ情報を調べる（ステップ４２０）。これによりもう１つののオペレーティング・システム（Ｊａｖａ仮想マシン）のバージョン番号が、クライアントにインストールされたバージョンに対応するかどうかが確認される（ブラウザはＤＭＩを使用して、このテストには必要ない属性は無視し、ＭＩＦファイルから選択された情報だけを取得できることに注意されたい）。所定基準が満足されない場合、アプレットの処理は終了し、アプレットがダウンロードされることはない。決定のために常時または条件に応じてユーザに提示される何らかの基準があってもよい。例えばアプレットが所定サイズより大きい場合、アプレットをダウンロードするのに必要な時間の長さを考慮して、操作を続けるかどうかユーザに確認を求めることができる（ステップ４２５）。所定基準が満足されたとすると（必要なユーザ確認があるとき）、アプレットがダウンロードされる（ステップ４３０）。この段階で通常はセキュリティのために別のテストを行うことができる（ステップ４３５）。例えばアプレットが正しいサイズかどうか確認でき、ＭＩＦファイルで指定された特定のバイト数が、アプレットそのものの内容と合致しているかどうかのチェックが行える。所定基準が再び満たされたとすると（ここでもステップ４４０で必要なユーザ確認があるかどうかによる）、アプレットを起動することができる（ステップ４４５）。ブラウザは後にクライアント・システムからアプレットを削除する場合、一般にはＭＩＦデータベースを更新する必要がある。

【００３８】図３、図４に示した処理については様々な変形が可能であることは理解されよう。例えばＭＩＦファイルは、所定基準に照らしたチェックを行うことなく、或いはユーザ確認を求めることなく、ただ監視を目的にＭＩＦデータベースにロードすることもできる（ステップ４１０）。通常これは、アプレットがダウンロードされた後に行われ（ステップ４３０）、そこでブラウザがＭＩＦファイルの内容とは無関係にアプレットを起動しようとする。またブラウザによっては、テストをユーザに明らかにすることなく、所定基準に対するテストを全て自動的に行うこともできる。もう１つ考えられるのは、ＭＩＦファイルはアプレットを実行するために必要な情報を保持してもよい。例えばアプレットは改変の可能性を小さくするため、暗号化された形式でダウンロードすることが検討されているが、その場合、ＭＩＦによりアプレットを解読するのに必要なアルゴリズムを指定することもできる。

【００３９】ＭＩＦファイルがＭＩＦデータベースにロードされるかどうか、そうなら、ブラウザがＭＩＦ情報をＭＩＦファイルそのものから直接、またはＭＩＦデータベースを使用して保持するかどうかは任意である。ＭＩＦをＭＩＦデータベースにロードするのは監視が目的なら便利であろう。またこれによりブラウザはＭＩを利用して、ＭＩＦファイルからＭＩＦ情報を取得することができる。これによりブラウザがＭＩＦファイルを解析して、様々な属性の値を取得する必要はなくなる。しかし、多数のアプレットがクライアント・マシンにダウンロードされ、実行され削除される前に、そこに一時的にしか存在しない場合、場合によっては、ＤＭＩとの対話に伴うオーバーヘッドを許容するよりも、ＭＩＦファイル自体からＭＩＦ情報を直接取得する方が望ましいかもしれない。

【００４０】ここで述べたＭＩＦ属性には、コードの大きさ、目的のオペレーティング・システムのバージョン番号等、既に標準のＤＭＩグループに存在するものがあり、他方、用いられる暗号の種類等、ユーザ定義グループにおく必要があるものもある。ＭＩＦに保持できる情報の種類については厳密な制限はないが、属性セットを好適にはほとんどの、或いは全てのブラウザが解釈できるＤＭＩ規格の一部として、予め定義しておく方が望ましいことは明らかである。またブラウザがこのセットの中の特定の属性を認識しない場合でも、可能なら関連アプレットを起動できることも望ましい（これは、ブラウザが特定のＨＴＭＬタグを認識しないが、一般には、所望のフォーマットに正確に一致はしなくとも、ページの情報を表示できる状況と多少とも似ている）。

【００４１】上の実施例で、ＭＩＦファイルはクライアント側で分析するためにその全体がダウンロードされる。これに代わる方法としては、アプレットに関する情報を取得するために、遠隔地にある関連ＭＩＦファイルに問い合わせをすることである。その場合、ＭＩＦファイルはＭＩＦデータベースに組み込まれている可能性が大きく、通常、ＭＩＦデータベースへのアクセスが与えられるＤＭＩを検出するために属性のＭＩＦとＭＩＦＢＡＳＥが用いられる。この方法は、基本的に図２に示したものと同じであるが、ここでＤＭＩ２８０はオペレーティング・システム２２０やブラウザ２１０とは別のマシン上に位置する点が異なる。

【００４２】ＤＭＩに対するこのリモート・アクセスを実現するメカニズムはいくつか考えられる。例えばＤＭＩ自体で、ＭＩを介したリモート・アクセスを直接サポートすることが求められるが、これを直接的に完全に実現することは今のところできない。その場合、処理は図４に示したようになるが、ステップ４１０は省略され、ステップ４１０でのＭＩＦ情報の取得はリモートアクセス機構を介して行われる。これに代えて、ブラウザに周知のＳＮＭＰ（Simple Network Management Protocol）を使用し、ＤＭＩにリモートで問い合わせをすることができる。ＤＭＩは既にこのメカニズムをサポートしているからである（ＳＮＭＰについては、Douglas E ComerとDavid L Stevensによる"Internetworking with TCP/IP、Volume2"、Prentice Hall、New Jersey、USA、1991を参照されたい）。もう１つ考えられるのは、ＳＮＭＰサポートをブラウザに組み込むことを回避するもので、ＭＩＦとＭＩＦＢＡＳＥにより同じマシン上の中間プログラムをＭＩＦデータベースとして識別することである。この中間プログラムは次に、ＨＴＴＰプロトコルで、ＭＩＦ情報に対する要求をブラウザから受け取る。要求では特定の所望属性が指定される。中間プログラムは、これらの要求をＤＭＩのＭＩに提示してから、応答をブラウザに再びＨＴＴＰプロトコルで返す。

【００４３】ＭＩＦ情報を遠隔地から直接使用する方が、最初にＭＩＦ情報をクライアントにコピーするよりも効率がよい場合があり、これは特にブラウザが認識するＭＩＦ属性が比較的少ない場合である。これによりＭＩＦファイル全体をネットワークで転送する必要がなくなるからである。もちろんこの方法ではブラウザが所望の監視情報を個別に記録する必要がある。

【００４４】先の説明では、アプレット情報の識別子はＭＩＦＢＡＳＥ、ＭＩＦと呼ばれる２つの要素に分けられているが、識別子は適切な名前で、適切な形で任意に表現できることは明らかである。またコードに関する情報はＭＩＦファイルにする必要はなく、ＳＮＭＰ管理情報ベース（ＭＩＢ）形式等、別の形式、或いはまたユーザが定義なり指定なりした他の形式であってもよい。更に例２の識別子はＡＰＰＬＥＴタグそのものの中に置かれるが、別のタグを＜ＡＰＰＬＥＴ＞、＜／ＡＰＰＬＥＴ＞の間に置き（すなわち＜ＰＡＲＡＭ＞タグと同様）、ここに識別子を保存してもよい。もちろん、前記のことを全て考慮した上で、ある１つのオプションの方へ標準化し、その後全てのブラウザ或いはほとんどのブラウザがこれをサポートできることが望ましい。

【００４５】更に、上に述べた実施例では、ＭＩＦファイルを調べて解釈するコードがブラウザに組み込まれているが、このコードはクライアント側に存在し、ブラウザから認識される別のプログラムまたはオブジェクトを形成することもできる。従ってＭＩＦファイルを取得する際、ブラウザはＭＩＦファイルをＭＩＦインタープリタ（プログラム）に渡す。インタープリタはそこでファイルを所望の方法で分析する。例えばインタープリタは単にＭＩＦファイルを表示するウィンドウを開き、また可能なら操作を続けるかどうかの確認を求め（図３のステップ３５０、３６０に対応）、或いは監視を目的に単にＭＩＦファイルをローカルのＭＩＦデータベースに追加するか、または所定基準に照らしてＭＩＦファイルをテストする。可能ならＭＩＦインタープリタはブラウザに応答を返して、アプレットの実行に進むかどうかを示す。

【００４６】これに代えて、ＭＩＦインタープリタは、Ｊａｖａ仮想マシン自体から参照することもできる。仮想マシンはアプレットを実行するため起動されると、最初、ＭＩＦインタープリタを実行し、例えば監視を目的にアプレットの使用状況を記録するか、またはインストールされた仮想マシンのバージョンがアプレットを実行するのに必要なものと対応するかどうか確認する。従って、ＭＩＦインタープリタは仮想マシンで一種のプリプロセッサとして、バイト・コード・バリデータ（validator）と同様の方法で（その直前に）働く。現在バリデータは最初に、実行のため仮想マシンに渡されたバイト・コードを確認する。この方法で仮想マシンは、好適には第３者のＭＩＦインタープリタ・プログラムが仮想マシンに結び付けられるようにフックを持つが、同時にＭＩＦインタープリタ機能を仮想マシンそのものに組み込むことも可能である。

【００４７】付け加えると、ここまで好適な実施例について、他の場合はインターネットに接続された従来型のコンピュータ・ワークステーションを中心に説明してきたが、本発明がこれに限定されないことは理解されよう。例えばコンピュータ・ワークステーションは、いわゆるネットワーク・コンピュータに置き換え、或いはネットワークを通してコードを受信できる他のハードウェア、例えばセット・トップ・ボックス、インターネットにアクセスできるテレビ等に置き換えることもできる。同様に関連するネットワークはインターネットでなくともよく、企業イントラネット（通常はＨＴＭＬ等のインターネット・プロトコルを使用するが、これはインター

ネット自体からは切り離されている）、或いは他のネットワークでもよい。更に好適な実施例でダウンロードされるコードは、Ｊａｖａバイト・コードを含むが、このコードは任意の言語でよく、例えばＪａｖａのように仮想マシンにより解釈される形でも、またクライアント・マシンのハードウェアで直接実行されるものでもよい。その場合、ダウンロードされるコードの目的オペレーティング・システムがクライアント・マシンにインストールされたものに一致するか確認するためにＭＩＦファイルを使用できる。

【００４８】まとめとして、本発明の構成に関して以下の事項を開示する。

【００４９】（１）ネットワークに接続されたコンピュータ側で実行されるよう、ネットワークを通して取得できるコードへの参照を含むデータの少なくとも１ページにアクセスできる、前記コンピュータを操作する方法であって、前記参照に関連付けられた、前記コードに関する情報の識別子を検出するステップと、前記コードに関する情報を取得するステップと、を含む、方法。

（２）前記コードを実行するかどうかを決定するために、前記情報を利用するステップを含む、前記（１）記載の方法。

（３）前記コンピュータのユーザに前記情報の少なくとも一部を提示することで、前記コードを実行するかどうかを前記ユーザが決定できるようにするステップを含む、前記（２）記載の方法。

（４）前記情報の少なくとも一部を所定基準と比較することで、前記コードを実行するかどうかを決定するステップを含む、前記（２）または（３）記載の方法。

（５）前記データの少なくとも１ページはＨＴＭＬ形式である、前記（１）乃至（４）のいずれかに記載の方法。

（６）前記参照はＡＰＰＬＥＴタグを含む、前記（５）記載の方法。

（７）前記識別子は前記ＡＰＰＬＥＴタグ内の属性を含む、前記（６）記載の方法。

（８）前記識別子は、前記情報が置かれるサーバとディレクトリを指定する第１属性と、前記情報を含むファイルの名前を指定する第２属性とを含む、前記（７）記載の方法。

（９）前記コンピュータは、Ｗｅｂブラウザを実行して前記データの少なくとも１ページにアクセスし、前記参照は汎用リソース・ロケータ（ＵＲＬ）の形である、前記（１）乃至（８）のいずれかに記載の方法。

（１０）前記識別子は汎用リソース・ロケータ（ＵＲＬ）の形である、前記（１）乃至（９）のいずれかに記載の方法。

（１１）前記情報は、管理情報形式（ＭＩＦ）に準拠した形で保存される、前記（１）乃至（１０）のいずれかに記載の方法。

（１２）前記情報はファイルに保存され、前記コードに関する情報を取得するステップは、前記ファイルを前記コンピュータにコピーするステップを含む、前記（１）乃至（１１）のいずれかに記載の方法。

（１３）前記情報は、管理情報形式（ＭＩＦ）に準拠した形で保存され、前記コンピュータ側のＭＩＦデータベースにロードされる、前記（１２）記載の方法。

（１４）ロードされたＭＩＦ情報にアクセスするため前記ＭＩＦデータベースに問い合わせをするステップを含む、前記（１３）記載の方法。

（１５）前記情報はリモート・データベースに保存され、前記情報を取得するステップは、ネットワークを通して前記データベースにリモートで問い合わせをするステップを含む、前記（１）乃至（１１）記載の方法。

（１６）前記情報は前記コードのサイズを含む、前記（１）乃至（１５）のいずれかに記載の方法。

（１７）ネットワークを通して前記コードを取得するため前記参照を使用するステップを含む、前記（１）乃至（１６）のいずれかに記載の方法。

（１８）取得されたコードを前記コンピュータ側で実行するステップを含む、前記（１７）記載の方法。

（１９）ネットワークに接続でき、コンピュータ側で実行するため、前記ネットワークを通して取得できるコードへの参照を含むデータの少なくとも１ページにアクセスするコンピュータ・システムであって、前記参照に関連付けられた、前記コードに関する情報の識別子を検出する手段と、前記コードに関する情報を取得する手段と、を含む、コンピュータ・システム。

（２０）前記コードを実行するかどうかを決定するために、前記情報を利用する手段を含む、前記（１９）記載のコンピュータ・システム。

（２１）前記コンピュータのユーザに前記情報の少なくとも一部を提示することで、前記コードを実行するかどうかを前記ユーザが決定できるようにする手段を含む、前記（２０）記載のコンピュータ・システム。

（２２）前記情報の少なくとも一部を所定基準と比較することで、前記コードを実行するかどうかを決定する手段を含む、前記（２０）または（２１）記載のコンピュータ・システム。

（２３）前記データの少なくとも１ページはＨＴＭＬ形式である、前記（１９）乃至（２２）のいずれかに記載のコンピュータ・システム。

（２４）前記参照はＡＰＰＬＥＴタグを含む、前記（２３）記載のコンピュータ・システム。

（２５）前記識別子は、前記ＡＰＰＬＥＴタグ内の属性を含む、前記（２４）記載のコンピュータ・システム。

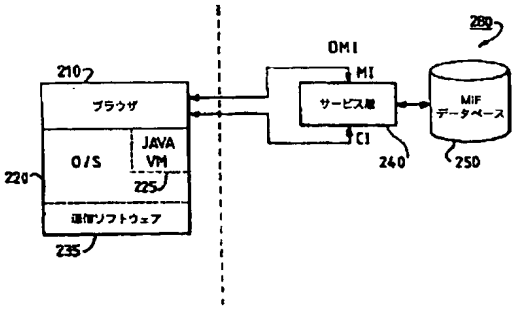
（２６）前記識別子は、前記情報が置かれるサーバとディレクトリを指定する第１属性と、前記情報を含むファイルの名前を指定する第２属性とを含む、前記（２５）記載のコンピュータ・システム。

（２７）前記コンピュータは、Ｗｅｂブラウザを実行して前記データの少なくとも１ページにアクセスし、前記参照は汎用リソース・ロケータ（ＵＲＬ）の形である、前記（１９）乃至（２６）のいずれかに記載のコンピュータ・システム。
（２８）前記識別子は汎用リソース・ロケータ（ＵＲＬ）の形である、前記（１９）乃至（２７）のいずれかに記載のコンピュータ・システム。
（２９）前記情報は、管理情報形式（ＭＩＦ）に準拠した形で保存される、前記（１９）乃至（２８）のいずれかに記載のコンピュータ・システム。
（３０）前記情報はファイルに保存され、前記コードに関する情報を取得する手段は、前記ファイルを前記コンピュータにコピーする手段を含む、前記（１９）乃至（２９）のいずれかに記載のコンピュータ・システム。
（３１）前記情報は、管理情報形式（ＭＩＦ）に準拠した形で保存され、前記コンピュータ側のＭＩＦデータベースにロードされる、前記（３０）記載のコンピュータ・システム。
（３２）ロードされたＭＩＦ情報にアクセスするため前記ＭＩＦデータベースに問い合わせをする手段を含む、前記（３１）記載のコンピュータ・システム。
（３３）前記情報はリモート・データベースに保存され、前記情報を取得する手段は、ネットワークを通して前記データベースにリモートで問い合わせをする手段を含む、前記（１９）乃至（２９）のいずれかに記載のコンピュータ・システム。
（３４）前記情報は前記コードのサイズを含む、前記（１９）乃至（３３）のいずれかに記載のコンピュータ・システム。
（３５）ネットワークを通して前記コードを取得するため前記参照を使用する手段を含む、前記（１９）乃至（３４）のいずれかに記載のコンピュータ・システム。
（３６）取得されたコードを前記コンピュータ側で実行する手段を含む、前記（３５）記載のコンピュータ・システム。
（３７）ネットワークに接続されたコンピュータによって実行され、前記コンピュータ側で実行するためにネットワークを通して取得できるコードへの参照を含むデータの少なくとも１ページにアクセスする、コンピュータ・プログラム・プロダクトであって、前記参照に関連付けられた、前記コードに関する情報の識別子を検出する手段と、前記コードに関する情報を取得する手段と、を含む、コンピュータ・プログラム・プロダクト。
（３８）前記コードを実行するかどうかを決定するために、前記情報を利用する手段を含む、前記（３７）記載のコンピュータ・プログラム・プロダクト。
（３９）前記コンピュータのユーザに前記情報の少なくとも一部を提示することで、前記コードを実行するかどうかを前記ユーザが決定できるようにする手段を含む、前記（３８）記載のコンピュータ・プログラム・プロダクト。
（４０）前記情報の少なくとも一部を所定基準と比較することで、前記コードを実行するかどうかを決定する手段を含む、前記（３８）または（３９）記載のコンピュータ・プログラム・プロダクト。
（４１）前記情報はファイルに保存され、前記コードに関する情報を取得する手段は、前記ファイルを前記コンピュータにコピーする手段を含む、前記（３７）乃至（４０）のいずれかに記載のコンピュータ・プログラム・プロダクト。
（４２）前記情報は、管理情報形式（ＭＩＦ）に準拠した形で保存され、前記コンピュータ側のＭＩＦデータベースにロードされる、前記（４１）記載のコンピュータ・プログラム・プロダクト。
（４３）ロードされたＭＩＦ情報にアクセスするため前記ＭＩＦデータベースに問い合わせをする手段を含む、前記（４２）記載のコンピュータ・プログラム・プロダクト。
（４４）前記情報はリモート・データベースに保存され、前記情報を取得する手段は、ネットワークを通して前記データベースにリモートで問い合わせをする手段を含む前記（３７）乃至（４０）のいずれかに記載のコンピュータ・プログラム・プロダクト。
（４５）ネットワークを通して前記コードを取得するため前記参照を使用する手段を含む、前記（３７）乃至（４４）のいずれかに記載のコンピュータ・プログラム・プロダクト。
（４６）取得されたコードを前記コンピュータ側で実行する手段を含む、前記（４５）記載のコンピュータ・プログラム・プロダクト。
（４７）インターネットを通して取得でき、コンピュータ側で実行されるアプレット・コードへの参照を含む、ワールド・ワイド・ウェブ上のデータの少なくとも１ページにアクセスするＷｅｂブラウザであって、前記参照に関連付けられた、前記アプレット・コードに関する情報の識別子を検出する手段と、前記コードに関する情報を取得する手段と、を含む、ブラウザ。
（４８）前記コードを実行するかどうかを決定するために、前記情報を利用する手段を含む、前記（４７）記載のブラウザ。
（４９）前記コンピュータのユーザに前記情報の少なくとも一部を提示することで、前記コードを実行するかどうかを前記ユーザが決定できるようにする手段を含む、前記（４７）記載のブラウザ。
（５０）前記情報の少なくとも一部を所定基準と比較することで、前記コードを実行するかどうかを決定する手段を含む、前記（４８）または（４９）記載のブラウザ。
（５１）前記参照はＡＰＰＬＥＴタグを含み、前記識別

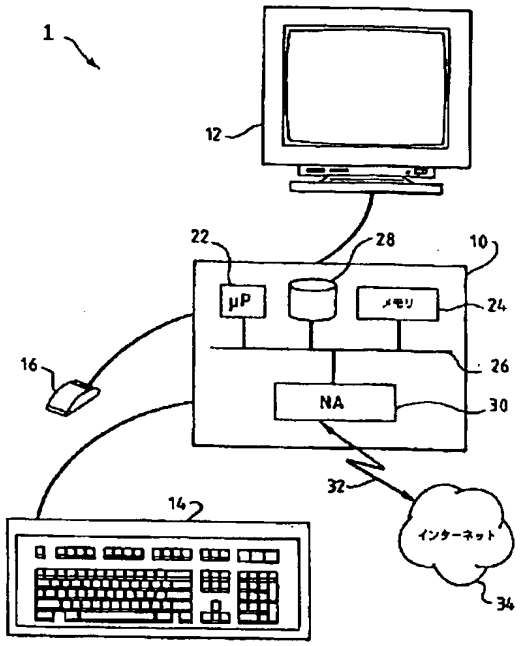
子は前記ＡＰＰＬＥＴタグ内の属性を含む、前記（４７）乃至（５０）のいずれかに記載のブラウザ。
（５２）前記識別子は、前記情報が置かれるサーバとディレクトリを指定する第１属性と、前記情報を含むファイルの名前を指定する第２属性とを含む、前記（５１）記載のブラウザ。
（５３）前記情報は、管理情報形式（ＭＩＦ）に準拠した形で保存される、前記（４７）乃至（５２）のいずれかに記載のブラウザ。
（５４）前記情報はファイルに保存され、前記コードに関する情報を取得する手段は、前記ファイルを前記コンピュータにコピーする手段を含む、前記（４７）乃至（５３）のいずれかに記載のブラウザ。
（５５）前記情報は、管理情報形式（ＭＩＦ）に準拠した形で保存され、前記コンピュータ側のＭＩＦデータベースにロードされる、前記（５４）記載のブラウザ。
（５６）ロードされたＭＩＦ情報にアクセスするため前記ＭＩＦデータベースに問い合わせをする手段を含む、前記（５５）記載のブラウザ。
（５７）前記情報はリモート・データベースに保存され、前記情報を取得する手段は、ネットワークを通して前記データベースにリモートで問い合わせをする手段を含む前記（４７）乃至（５３）のいずれかに記載のブラウザ。
（５８）ネットワークを通して前記コードを取得し、取得されたコードを前記コンピュータ側で実行する手段を含む、前記（４７）乃至（５７）のいずれかに記載のブラウザ。

【図面の簡単な説明】
【図１】コンピュータ・システムの概要図である。
【図２】図１のコンピュータ・システムの主なソフトウェア要素の概要図である。
【図３】本発明の第１実施例に従った処理のフローチャートを示す図である。
【図４】本発明の第２実施例に従った処理のフローチャートを示す図である。
【符号の説明】
１　コンピュータ・システム
１０　システム装置
１２　表示画面
１４　キーボード
１６　マウス
２２　マイクロプロセッサ
２４　半導体メモリ（ＲＯＭ／ＲＡＭ）
２６　バス
２８　ハード・ディスク・ドライブ
３０　ネットワーク・アダプタ・カード
３２　リンク
３４　インターネット
２１０　Ｗｅｂブラウザ
２２０　オペレーティング・システム
２２５　Ｊａｖａ仮想マシン（ＶＭ）
２３５　通信ソフトウェア
２４０　サービス層
２８０　ＭＩＦデータベース

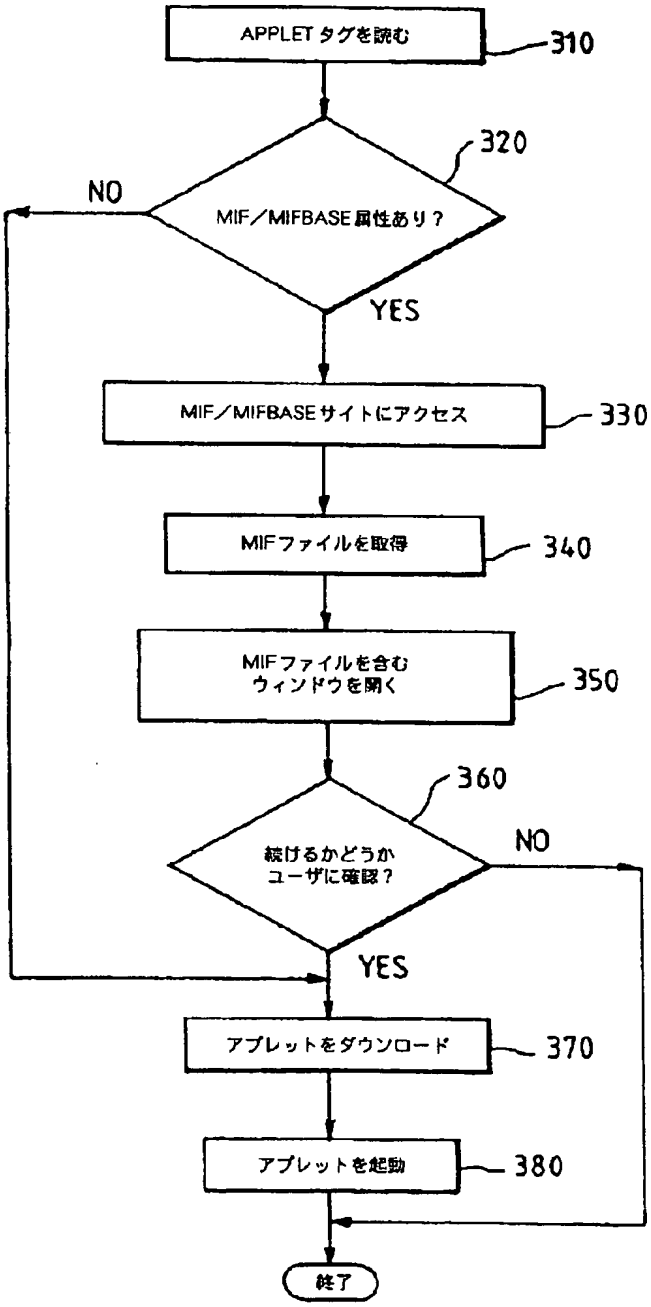
【図２】



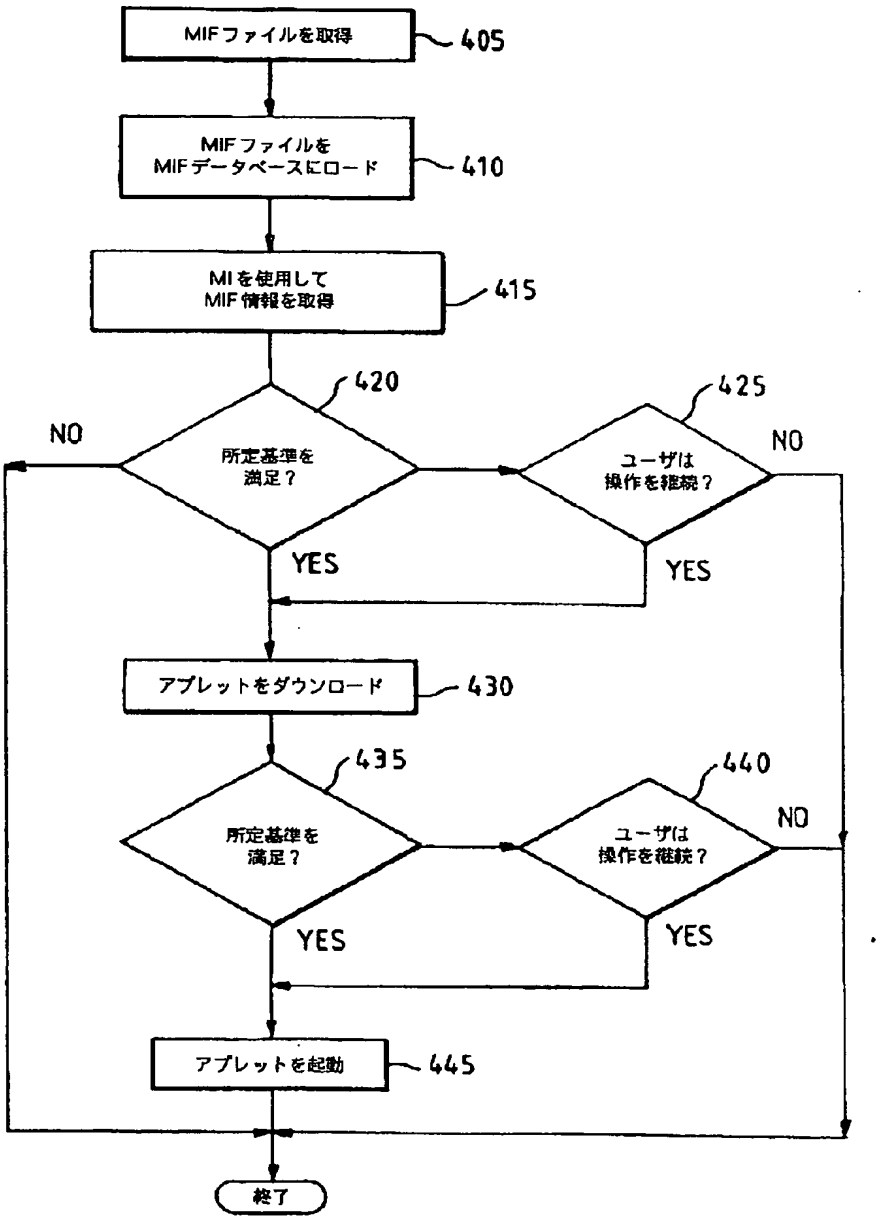
【図１】


1
12
22
28
10
μP
メモリ
24
26
16
NA
30
32
インターネット
14
34

【図３】


APPLETタグを読む
310
320
NO
MIF／MIFBASE属性あり？
YES
MIF／MIFBASEサイトにアクセス
330
MIFファイルを取得
340
MIFファイルを含む
ウィンドウを開く
350
360
続けるかどうか
ユーザに確認？
NO
YES
アプレットをダウンロード
370
アプレットを起動
380
終了

【図４】



フロントページの続き

(72)発明者　アラン・オギルビ
イギリス、エス・オー52　９エヌ・エイ
ハンプシャー、サウスハンプトン、ノース・バッデスレイ、シルバン・ドライブ
27